|  |  |  |
| --- | --- | --- |
| Name | type | description |
| <,<=,=,>,>= | comparator | Compare |
| +BC0+ | varset | Random selected unsatisfied clause |
| +NEG-GAIN+,  +POS-GAIN+,  +NET-GAIN+ | gaintype | Enumerated constant which represents negative gain, positive gain, and net gain, respectively |
| +WFF+ | varset | Entire formula |
| +T+ | varset | A random generated solution |
| +MAX-TRIES+ | var | limit |
| +MINB+ | varset | a stack that stores all variables with the minimum break in the selected falsified clause. |
| +BREAK+ | var | the minimum number of satisfied clauses that would become falsified by flipping x |
| +WP+ | var | Walking probability |
| +PROB+ | vecteur | (p1,p2,p3…pn) if x(i) has negative or positive variable in the unsatisfied clause, then p(i)=0.5. Else, p(i)= pi =pi +(1 -pi)exp(satnum/ m -1)K  T = (t 1 , t 2 , …, t n), ti =0 or 1(False or True).  Satnum is the number of satisfied clause |

|  |  |  |  |
| --- | --- | --- | --- |
| name | type | arguments | description |
| IF-SAF | varset | T variable set | If T satisfies then returns T |
| SOL | valset | Vs1, vs2 varset | Generates a random solution vs2 of vs1.Return vs2 |
| FOR | void | v1,v2 constant | Do sth from v1 to v2 |
| IF-RAND-LT | var | p float  v1, v2 variable | If a randomly generated ﬂoating point number is less than p, then return v1, else return v2 |
| VAR-RANDOM | var | vs varset | Select a variable in a random unsatisfied clause and flip  Ex.RAND-FLIP+US0+ |
| GET-VALUE | val | vs varset  g gaintype | Returns variable from varset vs which has the best value of speciﬁed gaintype g. E.g.,(GET-VAR +BC0+ +NEG-GAIN+) returns the variable with lowest negative gain from broken clause +BC0+. |
| NOTSAT | void | V operation | Do V until result is satisfied |
| IF-TABU | var | age integer  v1, v2 variable | If the age of variable v1 is less than age, then return v2; otherwise, return v1. |
| IF-VAR-COND | var | c comparator g gaintype n integer v1, v2 variable | Compute gain(v1,g), the gain of v1 according to gaintype g. Return v1 if the expression [gain(v1,g) c n] returns true; otherwise return v2. E.g., (IF-VAR-COND = 0 +NET-GAIN+ v1 v2) returns v1 if the net gain of v1 is equal to 0, and otherwise returns v2. |
| IF-MINB-FLIP | void | v operation | If break=0 then flip randomly in +MINB+, else do the operation |
| IF-VAR-COMPARE | void | c comparator  v1,v2 variable  o operation | If v1 c v2, do the operation |
| IF-PROM-FLIP | void |  | if there exist promising variables then greedily select the least  recently flipped promising variable |
| OLDER-VAR | var | V1,v2 variable | Least recently ﬂipped of the variables v1 and v2 |
| IF-NOT-MIN-AGE | var | vs varset  v1,v2 variable | If v1 does not have minimal age among variables in a varset vs, then return v1,else v2 |
| IF-WEIGHT | var | vs varset | 选择变元 x , 使得翻转这个变元前不满足子句权重之和 减去翻转后不满足子句权重之和的差达到最大.若这个差所 能达到的最大值小于或等于 0 , 则将当前所有不满足子句的权重增加1(Weighting), 退出当前的翻转循环;否则, 将该变元翻转, 调整当前指派 T(gr eedy). |
| INIT-WEIGHT | var | p value | Set all weights=1, set +PROB+ equal to p |

**/\*标红色的是我之前写的几个函数，觉得不是很合适，想办法把它们用其他的函数代替掉\*/**

**/\*标黄色的gain type是论文作者用的，我不是很明白这代表什么含义\*/**

1.GSAT() /\*每次寻找翻转后使得不满足字句减少最多的变元（这里我把NEG-GAIN理解为使得不满足字句减少最多）\*/

(FOR 1 +MAX-TRIES+

(SOL +UFF+ +T+)

(FOR 1 +MAX-STEP+

(IF-SAF +T+)

(GET-VALUE +T+ +NEG-GAIN+)))

1. GSAT2() /\*增加权重的概念,每次寻找翻转前后权重差最大的变元\*/

(INIT-WEIGHT)

(FOR 1 +MAX-TRIES+

(SOL +UFF+ +T+)

(FOR 1 +MAX-STEP+

(IF-SAT +T+)

(IF-WEIGHT)))

1. WalkSAT(p)/\*每次以p的概率翻转不满足字句里的变元，否则选择翻转后不满足字句减少最多的变元\*/

(SOL +UFF+ +T+)

(NOTSAT

(IF-RAND-LT p

(VAR-RANDOM +BC0+)

(GET-VALUE +BC0+ +NEG-GAIN+)))

2. WalkSAT2(p) /\*启发式算法论文里的，不明白里面的gain type表示什么意思\*/

(IF-VAR-COND = +NEG-GAIN+ 0

(GET-VAR +BC0+ +NEG-GAIN+)

(IF-RAND-LTE 0.5

(GET-VAR +BC0+ +NEG-GAIN+)

(VAR-RANDOM +BC0+)))

3.BreakWalkSAT(p) /\*大意是找到所有翻转后本来满足的字句变成不满足最少的变元，存在一个集合里。以1-p的概率翻转这个集合里的变元。要再改进下里面的函数\*/

IF-MINB-FLIP (

(IF-RAND-LT p

(VAR-RANDOM +BC0+)

(VAR-RANDOM +MINB+)))

4.Sparrow 2011/\*西安交大论文里提到的算法。要么随机翻转任意不满足字句里的变元，要么翻转目前为止已翻转次数最少，且使不满足字句减少最多的变元，要么根据权值函数翻转变元。现在我主要是对执行这三种情况的条件不太理解，比如walking probability=0.01的时候做第一种情况，但是这个walking probability是怎么求的？如果是自己设的，那么不是可能永远做第一步（只要等于0.01）或者永远不做第一步？\*/

FOR 1 +MAX-FLIPS+(

FOR 1 +MAX-STEP(

(IF-SAF +T+)

(IF-VAR-COMARE = +WP+ 0.01 (VAR-RANDOM +US0+))

(IF-PROM-FLIP)

5.Novelty+(0.5,0.01) (noise 0.5 and walk probability =0.01) /\*论文里用的

(IF-RAND-LT 0.01

(VAR-RANDOM +BC0+)

(IF-NOT-MIN-AGE +BC0+

(GET-VAR +BC0+ +NET-GAIN+)

(IF-RAND-LT 0.5

(GET-VAR2 +BC0+ +NET-GAIN+)

(GET-VAR +BC0+ +NET-GAIN+))))